Compiling Systemc

From GluCki

A SIMPLE PROGRAM TO PRINT HELLO WORLD

```
// All systemc modules should include systemc.h header file
   #include "systemc.h"
// Hello_world is module name
 SC_MODULE (hello_world) {
   SC_CTOR (hello_world) {
     // Nothing in constructor
   }
   void say_hello() {
     //Print "Hello World" to the console.
     cout << "Hello World.\n";
   }
 };

 // sc_main in top level function like in C++ main
 int sc_main(int argc, char* argv[]) {
   hello_world hello("HELLO");
   // Print the hello world
   hello.say_hello();
   return(0);
 }
```

Now to compile the above mentioned program, execute the following command:

```
g++ -I. -I$SYSTEMC_HOME/include -L. -L$SYSTEMC_HOME/lib-linux -o sim hel
```

On doing so you may encounter the following error:

```
: fatal error: systemc.h: No such file or directory compilation terminat
```

This is because you have not specified the path for value of **SYSTEMC_HOME** variable used above.

You can set the variable value using the following command (assuming you have installed systemc in **/usr/local/systemc-2.3.0**):

```
export SYSTEMC_HOME=/usr/local/systemc-2.3.0
```

You may still face the same error. This can be because your system may be 64 bit or your installation may not be proper. If you have 64 bit system, change the compiling command to:

```
g++ -I. -I$SYSTEMC_HOME/include -L. -L$SYSTEMC_HOME/lib-linux64 -o sim
```

Now if everything goes right, your code will compile successfully, but you may face another error, when you try to run the executable.

```
error while loading shared libraries: libsystemc-2.3.0.so: cannot open s
No such file or directory
```

This means that although the library has succesfully linked with executable (sim), at run time, the loader could not find the appropriate path for the library. The loader would look for the library in the default search path, prefixed by list of colon separated list of libraries supplied by LD_LIBRARY_PATH variable. When the loader could not find the described library, it reports such error.

To get rid of this error you can do the following things:

A) Set the value of **LD_LIBRARY_PATH** to the lib-linux directory in the systemc directory by using the following command:

```
export LD_LIBRARY_PATH=/usr/local/systemc-2.3.0/lib-linux
```

For the 64-bit system users, set the variable value as

```
export LD_LIBRARY_PATH=/usr/local/systemc-2.3.0/lib-linux64
```

To save yourself from setting the above mentioned variables again and again, you can put them in .bashrc which is present in your home directory.

B) Alternatively, you can set -Wl,-rpath flags at compilation. These flags hard-code the library path into the executable. So it knows where to look for the library. So your final compilation command would look something like this:

```
g++ -I. -I$SYSTEMC_HOME/include -L. -L$SYSTEMC_HOME/lib-linux -Wl,-rpath
```

For 64-bit systems:

```
g++ -I. -I$SYSTEMC_HOME/include -L. -L$SYSTEMC_HOME/lib-linux64 -Wl,-rpa
```

This seems a lot to compile a simple "Hello World" program. To save yourself from typing all of this and pressing upward arrow to remove a simple compile time error, you can create a makefile. Or just download from here (https://gist.github.com/rahul-raturi/8c2914caca8d6d75bc28).

Paste this file in the directory where your "hello.cpp" is. Just type **make** and you are good to go. An executable named **sim** will be created in that directory.

Note: In order to compile again you will have to first type **make clean**.

Retrieved from "http://glug.nith.ac.in/wiki/index.php?title=Compiling_Systemc&oldid=571"

- This page was last modified on 25 February 2015, at 13:43.
- This page has been accessed 3,030 times.
-